Amendments to the Claims

The following listing of claims will replace all prior versions and listings of claims in the application.

1. (Withdrawn) An interrupt manager for use in a distributed control system, the interrupt manager comprising:

circuitry that:

(i) receives interrupt signals including a current interrupt;

(ii) determines whether the current interrupt can be processed without delaying processing of a non-interrupt task beyond a predetermined time; and

(iii) inhibits, at least temporarily, processing of the current interrupt when it is determined that the processing of the current interrupt would delay the processing of the non-interrupt task beyond the predetermined time.

2. (Withdrawn) The interrupt manager of claim 1, wherein the circuitry determines whether the current interrupt can be processed without delaying the processing of the non-interrupt task beyond the predetermined time by determining whether a total number of interrupts including at least one of the current interrupt, a recently-performed interrupt and a pending interrupt would exceed a maximum number of interrupts.

3. (Withdrawn) The interrupt manager of claim 2, wherein the maximum number is associated with a time interval and represents a maximum number of interrupts that can be performed within the time interval.

4. (Withdrawn) The interrupt manager of claim 3, wherein the total number of interrupts includes, in addition to the current interrupt, any interrupts that have been received and not yet processed.

5. (Withdrawn) The interrupt manager of claim 3, wherein the total number of interrupts includes, in addition to the current interrupt, all interrupts that have been received since a first time.

6. (Withdrawn) The interrupt manager of claim 1, wherein the circuitry determines whether the current interrupt can be processed without delaying the processing of the non-interrupt task beyond the predetermined time by determining whether the processing of the current interrupt could be completed within an interrupt window.

7. (Withdrawn) The interrupt manager of claim 6, wherein the interrupt window is refreshed upon expirations of window periods.

8. (Withdrawn) The interrupt manager of claim 1, wherein the circuitry determines whether the current interrupt can be processed without delaying the processing of the non-interrupt task beyond the predetermined time by determining whether the processing of the current interrupt could begin within an interrupt window.

9. (Withdrawn) The interrupt manager of claim 1, wherein the circuitry inhibits the processing of the current interrupt by masking the current interrupt.

10. (Withdrawn) The interrupt manager of claim 1, wherein the circuitry determines whether the current interrupt can be processed without delaying the processing of the non-interrupt task beyond the predetermined time by comparing a first priority associated with the current interrupt with a second priority of the non-interrupt task.

11. (Withdrawn) The interrupt manager of claim 10, wherein the first priority is that of a proxy task generated in response to the receiving of the current interrupt based upon scheduling data of a message causing the current interrupt.

12. (Withdrawn) The interrupt manager of claim 1, wherein the circuitry temporarily inhibits the processing of the current interrupt by placing information relating to the current interrupt in a later position in a queue.

13. (Withdrawn) The interrupt manager of claim 1, wherein the circuitry determines whether current interrupt can be processed without delaying the processing of the non-interrupt task beyond the predetermined time by determining whether at least one completion timing constraint associated with the non-interrupt task would be violated if the processing of the current interrupt occurred.

14. (Withdrawn) A method of handling interrupts for use with a processor in a distributed control system, the method comprising:

receiving a current interrupt signal;

determining whether processing of the current interrupt signal would delay processing of a non-interrupt task beyond a predetermined time; and

inhibiting, at least temporarily, the processing of the current interrupt signal when it is determined that the processing would delay the processing of the non-interrupt task beyond the predetermined time.

15. (Withdrawn) The method of claim 14, further comprising delaying the processing of the current interrupt signal to a later time if it is determined that the processing would delay the processing of the non-interrupt task beyond the predetermined time.

16. (Withdrawn) The method of claim 14, wherein the determining includes:

comparing a total number of interrupt signals including at least the current interrupt signal with a maximum number of interrupt signals.

17. (Withdrawn) The method of claim 14, wherein the determining includes determining whether at least one of: the processing of the current interrupt signal can be

begun within a current time window; and the processing of the current interrupt signal can be completed within the current window.

18. (Canceled)

19. (Withdrawn) A method of scheduling messages being transmitted on a network among spatially-distributed control components of a distributed control system, the method comprising:

receiving a message;

receiving a relative timing constraint concerning the message, wherein the relative timing constraint is indicative of an amount of time; and

inserting the message into a queue at a location that is a function of the relative timing constraint.

20. (Canceled)

21. (Withdrawn) The method of claim 19, wherein the location is also a function of a priority associated with the message and of an absolute timing constraint concerning the message.

22. (Withdrawn) The method of claim 21, wherein the absolute timing constraint is a particular time.

23. (Withdrawn) The method of claim 19, wherein the inserting of the message into the queue is governed by a message scheduler implemented by a processor executing a portion of a distributed operating system providing respective portions of an overall completion timing constraint of a communication circuit to each of a plurality of application programs, the respective portions setting respective deadlines for the application programs.

24. (Currently Amended) A method of coordinating a new control application program with other control application programs being performed on a distributed real-time operating system, wherein the distributed real-time operating system is for use with a control system having spatially separated industrial controllers~~control hardware nodes~~, each industrial controller~~node~~ having at least one resource, and the control system further including a network connecting the industrial controllers~~a plurality of hardware nodes~~, the method comprising:

(a) executing the real-time operating system on at least one processor of the industrial controllers~~hardware nodes~~;

~~(b) managing at least one resource from at least a first and a second, spatially separated control hardware node with the real-time operating system;~~

~~(c)~~(b) receiving the new control application to be registered with the real-time operating system;

~~(d) matching control hardware resources required by the new control application program from a resource list to resources on at least the first and the second spatially separated control hardware nodes;~~

~~(e)~~(c) identifying a fixed time interval ~~associated with the new control application program~~ for completing execution of at least a portion of the new control application program;

(d) identifying at least a first industrial controller and a second industrial controller on which the new control application program will execute;

(e) identifying a portion of the network connecting the first and second industrial controllers as required by the new control application program;

~~(f) allocating the new control application program to the identified control hardware resources;~~

~~(g) allocating portions of the fixed time interval associated with the new control application program to each identified control hardware resource; and~~

~~(h)~~(f) ~~determining whether the allocated portions of~~verifying that the new control application program can be executed within the ~~portion of the~~ fixed time interval ~~allocated to each identified control hardware resource~~using the first industrial controller, the second industrial controller, and the portion of the network connecting the first and second industrial controllers while requirements of the other control application programs also are met[[.]]; and

(g) allocating the new control application program to the identified control hardware resources if the new control application program can be executed within the fixed time interval.

25. (Canceled)

26. (Original) The method of claim 24, further comprising:

collecting statistics regarding a usage of the control hardware resources as the new control application program and other control application programs are being performed; and

optimizing the usage of the control hardware resources based at least in part upon the collected statistics.

27. (Currently amended) A method of operating an application program on a distributed control system having a plurality of industrial controllers~~nodes~~ wherein each industrial controller~~node~~ includes a memory device, a processor, and a communication means, the method comprising:

executing a real-time operating system on at least one processor of the industrial controllers~~nodes~~;

maintaining a topology map identifying the network connections between each of the industrial controllers;

maintaining a resource list identifying the utilization of the memory device, the processor, and the communication means of each industrial controller and the utilization of the network connections between each of the industrial controllers;

~~managing at least one resource from at least a first and a second node with the real-time operating system;~~

~~receiving high-level requirements concerning the application program;~~

~~determining low-level requirements based upon the high-level requirements;~~

obtaining a fixed completion time requirement for the application program;

identifying at least a first and a second industrial controller from the topology map, each industrial controller executing at least a portion of the application program;

~~allocating each of the high-level requirements and the low-level requirements to at least one of the nodes;~~

allocating at least a portion of the fixed completion time to each industrial controller identified from the topology map~~node having been allocated one of the requirements~~;

determining ~~at each node to which a requirement was allocated~~ whether the portion of the application program executing on each industrial controller identified from the topology map~~allocated requirements~~ may be completed within the portion of the fixed completion time allocated to that industrial controller~~node~~ based on the utilization of the industrial controller and the utilization of the network connections between the industrial controllers; and

operating the application program in accordance with the allocated requirements if each industrial controller can complete the portion of the application program executing thereon within the allocated portion of the fixed completion time.

28. (Currently amended) The method of claim 27, wherein each portion of the application program~~the high-level requirements~~ includes at least one of a hardware requirement, a completion-timing constraint, a message size, an inter-arrival period, a need for remote system services, and a type of priority~~, and~~

~~wherein the low-level requirements include at least one of an amount of memory, a network bandwidth, and a processor bandwidth~~.

29.-34. (Canceled)

35. (New) The method of claim 24 wherein the real time operating system maintains a topology map and wherein the step of identifying at least the first industrial

controller and the second industrial controller is based at least in part on the network connections between the first and the second industrial controllers.

36. (New) The method of claim 28 wherein each portion of the application program further requires at least one of an amount of memory, a network bandwidth, and a processor bandwidth.

37. (New) The method of claim 27 wherein a portion of the fixed completion time is allocated to the network connections between the first and the second industrial controllers.

38. (New) The method of claim 24 wherein the first industrial controller is connected to a first control signal of a controlled process, the second industrial controller is connected to a second control signal from the controlled process, and a portion of the new control application program utilizes both the first and the second control signals wherein the step of identifying at least a first industrial controller and a second industrial controller identifies the first and the second industrial controllers connected to the first and the second control signals.